Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1.   (Currently Amended) A method, in a data processing system, for resource allocation of a plurality of tasks carrying penalties based on their completion time, the method comprising:

assigning the plurality of tasks to one or more resources, wherein each job of a plurality of jobs includes multiple tasks ordered by chain precedence, wherein the plurality of jobs include active jobs and queued jobs, and wherein the plurality of tasks are associated with the active jobs; and

assigning start times for the plurality of tasks such that expected penalties for completion times of the plurality of tasks are minimized, wherein a schedule is implemented for the active jobs based on a best solution ~~the expected penalties are minimized by repeatedly assigning tasks and reevaluating start times for the plurality of tasks based on a plurality of predictable potential next events~~;

wherein the assigning steps are performed by a scheduling controller, wherein the scheduling controller responds synchronously to each event in response to receiving an event, and wherein the scheduling controller comprises a preemption module, a greedy module, a randomized module, a thinker module, an interrupt module, and a sleep module;

wherein the preemptive module performs assignments and reassignments for the plurality of jobs in response to receiving a new or changed job with a priority level that is greater than any of the active jobs;

wherein the greedy module performs immediate assignments and reassignments for the plurality of jobs based on expected execution times of tasks by executing an initial algorithm to form a preliminary solution for scheduling the plurality of jobs;

wherein the randomized module performs random assignments and reassignments for the plurality of jobs by executing a randomized algorithm and retains a random seed value that generates a current best random solution for scheduling the plurality of jobs using the randomized algorithm, wherein the random seed value is initialized to zero to indicate that the preliminary solution is the current best random solution, and wherein the random seed value is updated in response to executing the randomized algorithm additional times and finding a better solution than the current best random solution;

wherein the thinker module assigns a time period to an allocated think time partition in a time slot for each active job, wherein each time slot of a plurality of time slots comprises a plurality of allocated think time partitions corresponding to the active jobs, wherein the thinker

module first executes the greedy module and then repeatedly executes the randomized module for a duration of the time period assigned to the allocated thinking time partition to determine a probable best solution for scheduling the plurality of jobs, and wherein the probable best solution is calculated for each allocated thinking time partition;

wherein the interrupt module halts the execution of other modules;

wherein the sleep module causes the scheduling controller to wait;

~~allocating think time for each problem instance variant of the plurality of predictable potential next events into separate think time partitions within each time slot for determining a best solution for each problem instance variant of the plurality of the predictable potential next events, wherein an amount of think time is calculated for each problem instance variant of the plurality of the predictable potential next events, wherein each time slot is divided into a plurality of separate think time partitions, wherein allocating think time for each problem instance variant of the plurality of predictable potential next events includes:~~

~~reserving a first amount of time for performing an initial algorithm;~~

~~allocating a second amount of time for performing a randomized algorithm, wherein the randomized algorithm is a next algorithm;~~

~~during each allocated think time partition, allocating resources for a problem instance variant of a predicted next event at a predicted time at which the predicted next event may occur, wherein allocating the resources for the problem instance variant of the predicted next event includes:~~

~~executing the initial algorithm to form a preliminary solution;~~

~~recording a seed value of zero to indicate that a current solution is the preliminary solution; and~~

~~repeatedly executing the randomized algorithm until an event occurs or the second amount of time expires;~~

~~responsive to the randomized algorithm forming a solution that is better than a previous solution, updating the seed value; and~~

~~assigning resources for queued tasks based upon an actual next event and an actual time of occurrence, wherein assigning the resources for the queued tasks includes:~~

~~executing an algorithm that produced the best solution and assigning the resources based on results of the algorithm, wherein the algorithm is one of the initial algorithm and the randomized algorithm~~

determining whether the best solution was found using the initial algorithm or the randomized algorithm;

responsive to the best solution being found using the initial algorithm, executing the initial algorithm and assigning resources based on results of the initial algorithm; and

responsive to the best solution being found using the randomized algorithm, executing the randomized algorithm using the random seed value and assigning resources based on results of the randomized algorithm.

2.-8.   (Canceled)

9.   (Original) The method of claim 8, further comprising:
          assigning only immediately starting tasks.

10.   (Currently Amended) The method of claim 1, wherein [[an]] the event is one of a job arrival, a task completion, a data change arrival, a managerial schedule request, and a termination request.

11.-24. (Canceled)